CERTIFICATION OF TRANSLATION

I, ***Moung-kyo Kim***, an employee of Y.P. LEE, MOCK & PARTNERS of Koryo Building, 1575-1 Seocho-dong, Seocho-gu, Seoul, Republic of Korea 137-875, hereby declare under penalty of perjury that I understand the Korean language and the English language; that I am fully capable of translating from Korean to English and vice versa; and that, to the best of my knowledge and belief, the statement in the English language in the attached translation of ***Korean Patent Application No. 10-2006-0012728*** consisting of 8 pages, have the same meanings as the statements in the Korean language in the original document, a copy of which I have examined.

Signed this 1st day of December 2006

Moung kyo Kim

FIG. 1

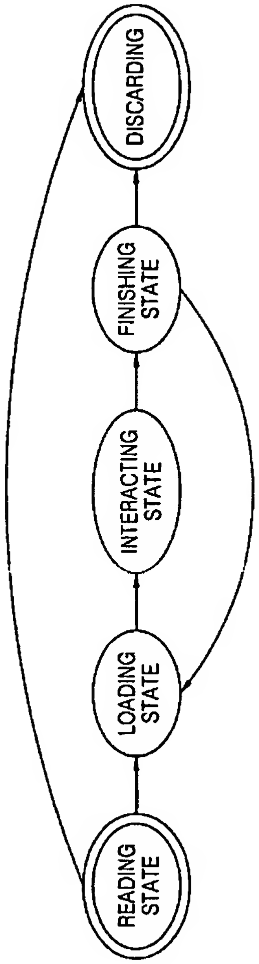

READING STATE
LOADING STATE
INTERACTING STATE
FINISHING STATE
DISCARDING

FIG. 2

A. Example of DOM–core module

```
<?xml version="1.0" ?>
<!—This is first comment —>
<!DOCTYPE html PUBLIC "-//DVD//DTD XHTML DVD-HTML 1.0//EN"
"http://www.dvdforum.org/enav/dtd/dvdhtml-1-0.dtd">
<html>
<head>
<title>DOM-core sample</title>
<script type="text/ecmascript">
alert("script code is in head tag")
function load_handler()
{
   alert("Hello, DVD-HTML")
   var strMsg="";
   strMsg+= document.documentElement.nodeName+"\n";
   strMsg+= document.documentElement.attributes.item(0).nodeName+"\n";
   var root= document.documentElement;
   for(var i=0;i<root.childNodes.length;i++)
   strMsg += "\t"+root.childNodes.item(i).nodeName+"\n";
   strMsg+= document.nodeName;
   alert(strMsg)
   alert(root.childNodes.item(0).childNodes.length);

alert(root.childNodes.item(0).childNodes.item(1).hasChildNodes());
}
function unload_handler()
{
   alert("good bye world")
}
</script>
</head>
<body onload="load_handler();" onunload="unload_handler();">
<script type="text/ecmacript">
alert("  before body text"  )
</script>
<p>body text</p>
<script type="text/ecmascript">
<![CDATA[
alert("after body text")
]]>
</script>
</body>
</html>
```

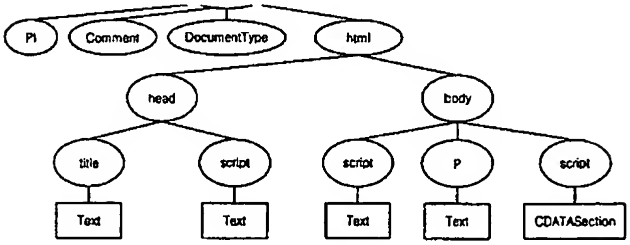
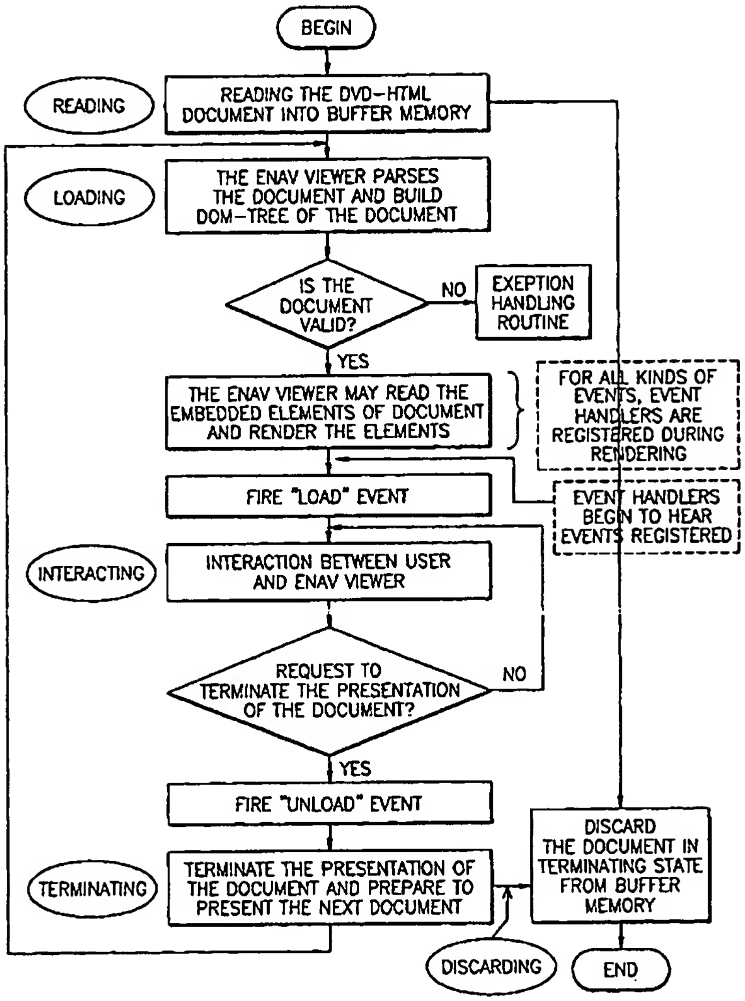


FIG. 3


BEGIN
READING
READING THE DVD-HTML DOCUMENT INTO BUFFER MEMORY
LOADING
THE ENAV VIEWER PARSES THE DOCUMENT AND BUILD DOM-TREE OF THE DOCUMENT
IS THE DOCUMENT VALID?
NO
EXEPTION HANDLING ROUTINE
YES
THE ENAV VIEWER MAY READ THE EMBEDDED ELEMENTS OF DOCUMENT AND RENDER THE ELEMENTS
FOR ALL KINDS OF EVENTS, EVENT HANDLERS ARE REGISTERED DURING RENDERING
FIRE "LOAD" EVENT
EVENT HANDLERS BEGIN TO HEAR EVENTS REGISTERED
INTERACTING
INTERACTION BETWEEN USER AND ENAV VIEWER
REQUEST TO TERMINATE THE PRESENTATION OF THE DOCUMENT?
NO
YES
FIRE "UNLOAD" EVENT
TERMINATING
TERMINATE THE PRESENTATION OF THE DOCUMENT AND PREPARE TO PRESENT THE NEXT DOCUMENT
DISCARD THE DOCUMENT IN TERMINATING STATE FROM BUFFER MEMORY
DISCARDING
END

ABSTRACT

[Abstract of the Disclosure]

Provided is a method such as interpreting a markup document and displaying on a monitor are provided for determining a lifecycle of a document and providing a monitor displaying compatibility. The method for interpreting a markup language in an optical recording medium includes storing the markup language in a cache memory of a reproducing apparatus and leading so that a resource related to the document is generated into an item in the memory, interpreting the markup language to organize a tree having a document object model format, rendering and loading so that a load event is generated to a script interpreting engine, interacting wherein user interaction is performed about a document loaded on a monitor of the reproducing apparatus, terminating wherein the document is finished from the monitor when the displayed document should be disappeared and discarding wherein a document remained in the cache memory is removed.

[Representative Drawing]

FIG. 1

SPECIFICATION

[Title of the Invention]

APPARATUS AND METHOD FOR MANAGING DOCUMENTS ON OPTICAL STORAGE MEDIUM BY INTEPRETING MARKUP LANGUAGE BY STAGES

[Brief Description of the Drawings]

FIG. 1 is a state view illustrating a method of interpreting markup language on an optical storage;

FIG. 2 is a flowchart illustrating a method of interpreting markup language on an optical storage; and

FIG. 3 is a diagram illustrating an example of a DOM tree generating code and a DOM tree generated according to the DOM tree.

[Detailed Description of the Invention]

[Object of the Invention]

[Technical Field of the Invention and Related Art prior to the Invention]

In an apparatus and method of interpreting a markup language document in an optical and displaying on a monitor, P2001-0065393 (title: optical recording medium recording preload information, reproducing method and apparatus) has an introducing compatibility problem. That is, it is unclear to displaying on the monitor, to interact with a user, finishing displaying and removing in a memory after preloading, and thus a disk can not be compatible with other disks. For example, it is unclear whether HTML event (load) is generated during displaying a document or after finishing displaying. In addition, it is unclear whether the document used in the memory is removed when the memory is returned after finishing displaying or not. Accordingly, methods such as interpreting a markup document and displaying on a monitor are provided for determining a lifecycle of a document and providing a monitor displaying compatibility.

[Technical Goal of the Invention]

The present invention provides methods such as interpreting a markup document and displaying on a monitor are provided for determining a lifecycle of a document and providing a monitor displaying compatibility.

[Structure and Operation of the Invention]

A browser embodied in an apparatus interpreting a markup language document interprets a markup document according to a detailed flowchart of FIG. 2. Periods of staying of a document in a reproducing apparatus is shown as follows according to a state changing view (FIG. 1).

1) reading state: In this state, a markup language lifts a markup document to a cache memory. Accordingly, a resource related to the document is generated into an item (context or on-memory item) in a memory.

2) loading state: A markup language having XML language format is interpreted to organize a tree having a document object model format. Then a rendering is finished on a monitor. A script interpret engine in the browser generates a load event. Here, the meaning of interpreting the markup language is syntax-checking whether a code context of a document is right, and document type definition (DTD)-checking whether meanings are right.

3) interacting state: In this state, a user pushes all about and scrolls a rendered document of a browser. That is, a displayed context of the document is changed according to an interaction with a user and an interaction between DVD AV reproducing engine and ENAV viewer (or presentation engine). When the basic document displayed on a monitor is moved to other document, upload is generated into a script interpreter engine in the browser.

4) terminating state: In this state, displaying of a document is finished, and then just a system memory is remained.

5) discarding state: In this state, a document, which is remained in a system and is not displayed, is removed in a memory. When the document is continued in this state, on-memory information, which can manage a corresponding document is not remained in a memory, is not remained any more.

[Effect of the Invention]

A browser embodied in an apparatus for interpreting a markup language document in an optical disk determines a document life cycle treating a document to provide a monitor display compatibility.

What is claimed is:

1. A method for interpreting a markup language in an optical recording medium, the method comprising:

storing the markup language in a cache memory of a reproducing apparatus and leading so that a resource related to the document is generated into an item in the memory;

interpreting the markup language to organize a tree having a document object model format, rendering and loading so that a load event is generated to a script interpreting engine;

interacting wherein user interaction is performed about a document loaded on a monitor of the reproducing apparatus;

terminating wherein the document is finished from the monitor when the displayed document should be disappeared; and

discarding wherein a document remained in the cache memory is removed.

2. An apparatus for interpreting a markup language in an optical recording medium, the apparatus comprising:

reading unit storing the markup language in a cache memory of a reproducing apparatus and leading so that a resource related to the document is generated into an item in the memory;

loading unit interpreting the markup language to organize a tree having a document object model format, rendering and loading so that a load event is generated to a script interpreting engine;

interacting unit performing interacting about a document loaded on a monitor of the reproducing apparatus;

terminating unit finishing the document from the monitor when the displayed document should be disappeared; and

discarding unit removing a document remained in the cache memory.

FIG. 1

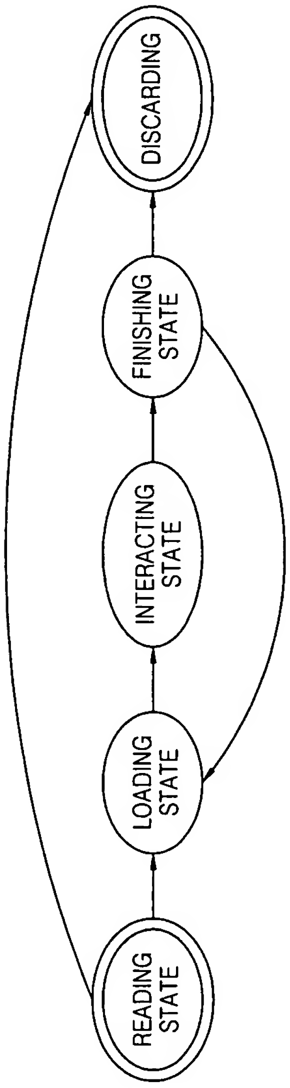

READING STATE
LOADING STATE
INTERACTING STATE
FINISHING STATE
DISCARDING

FIG. 2

A. Example of DOM-core module

```
<?xml version="1.0" ?>
<!—This is first comment -->
<!DOCTYPE html PUBLIC "-//DVD//DTD XHTML DVD-HTML 1.0//EN"
"http://www.dvdforum.org/enav/dtd/dvdhtml-1-0.dtd">
<html>
<head>
<title>DOM-core sample</title>
<script type="text/ecmascript">
alert("script code is in head tag")
function load_handler()
{
   alert("Hello, DVD-HTML")
   var strMsg="";
   strMsg+= document.documentElement.nodeName+"\n";
   strMsg+= document.documentElement.attributes.item(0).nodeName+"\n";
   var root= document.documentElement;
   for(var i=0;i<root.childNodes.length;i++)
   strMsg += "\t"+root.childNodes.item(i).nodeName+"\n";
   strMsg+= document.nodeName;
   alert(strMsg)
   alert(root.childNodes.item(0).childNodes.length);

alert(root.childNodes.item(0).childNodes.item(1).hasChildNodes());
}
function unload_handler()
{
   alert("good bye world")
}
</script>
</head>
<body onload="load_handler();" onunload="unload_handler();">
<script type="text/ecmacript">
alert("  before body text"  )
</script>
<p>body text</p>
<script type="text/ecmascript">
<![CDATA[
alert("after body text")
]]>
</script>
</body>
</html>
```

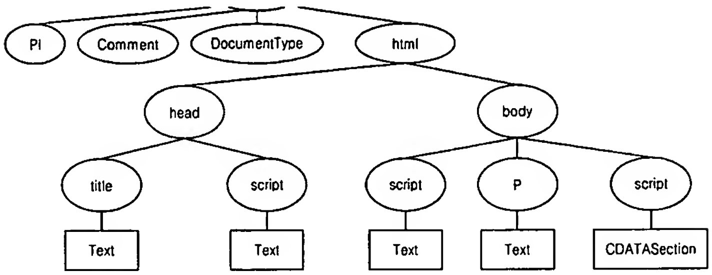
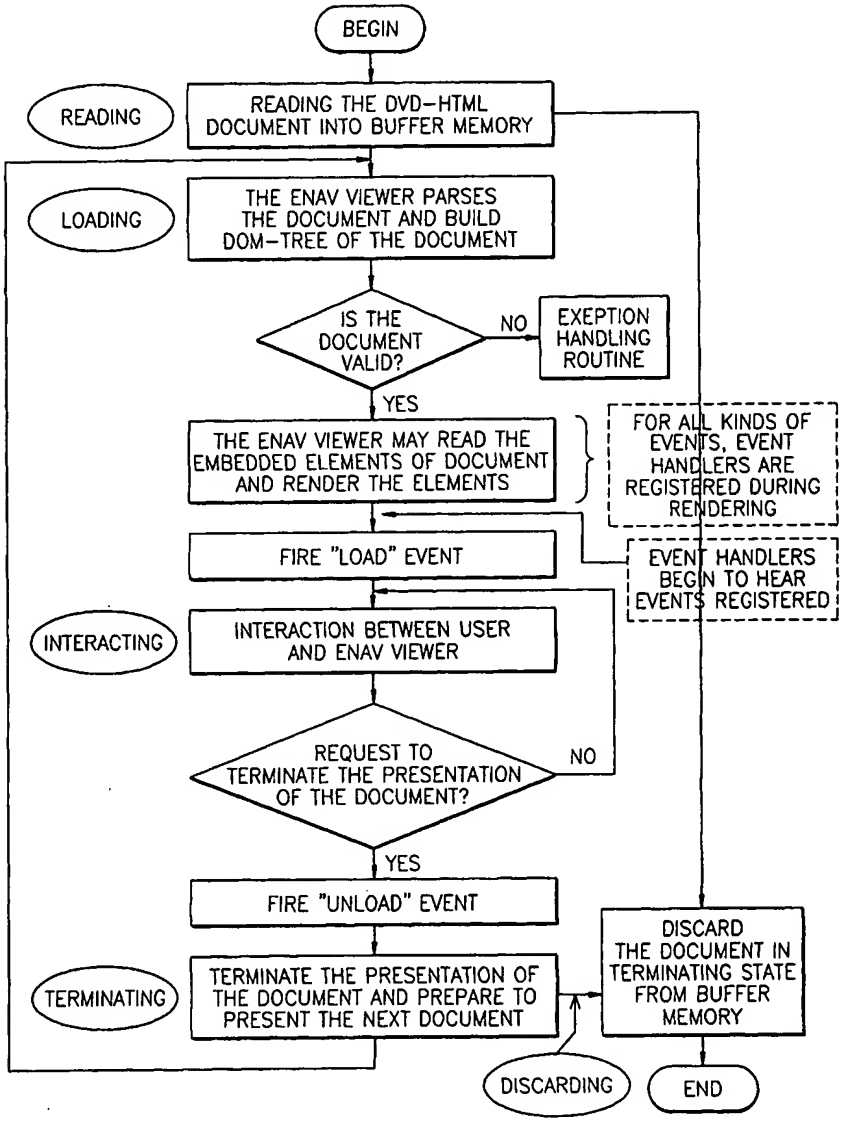


FIG. 3



BEGIN
READING
READING THE DVD-HTML DOCUMENT INTO BUFFER MEMORY
LOADING
THE ENAV VIEWER PARSES THE DOCUMENT AND BUILD DOM-TREE OF THE DOCUMENT
IS THE DOCUMENT VALID?
NO
EXEPTION HANDLING ROUTINE
YES
THE ENAV VIEWER MAY READ THE EMBEDDED ELEMENTS OF DOCUMENT AND RENDER THE ELEMENTS
FOR ALL KINDS OF EVENTS, EVENT HANDLERS ARE REGISTERED DURING RENDERING
FIRE "LOAD" EVENT
EVENT HANDLERS BEGIN TO HEAR EVENTS REGISTERED
INTERACTING
INTERACTION BETWEEN USER AND ENAV VIEWER
REQUEST TO TERMINATE THE PRESENTATION OF THE DOCUMENT?
NO
YES
FIRE "UNLOAD" EVENT
TERMINATING
TERMINATE THE PRESENTATION OF THE DOCUMENT AND PREPARE TO PRESENT THE NEXT DOCUMENT
DISCARD THE DOCUMENT IN TERMINATING STATE FROM BUFFER MEMORY
DISCARDING
END